DRAFT – 03/24/04

GAMMA-RAY LARGE AREA SPACE TELESCOPE (GLAST)

DATABASE FORMAT CONTROL DOCUMENT

March 24, 2004

GODDARD SPACE FLIGHT CENTER
GREENBELT, MARYLAND

GAMMA-RAY LARGE AREA SPACE TELESCOPE (GLAST) PROJECT

DATABASE FORMAT CONTROL DOCUMENT (DFCD)

Draft – March 24, 2004

NASA Goddard Space Flight Center

Greenbelt, Maryland

DOCUMENT APPROVAL
GLAST DATABASE FORMAT CONTROL DOCUMENT
March 2004

Formatted by:

Ernest V. Canevari ASRC Aerospace System Engineer	Date

Concurrence:

Dennis Small MOC	Date	Doug Spiegel MOC	Date
Jay Norris GSSC	Date		
Dave Lung LAT ISOC	Date	Mark Davis Spectrum Astro	Date
William Paciesas GBM IOC	Date		Date

Approved by:

Ken Lehtonen GLAST Ground System/Ops Manager	Date	Kevin Grady GLAST Project Manager	Date

REVISION STATUS

The GLAST Project Control Board (CCB) controls this document. Proposed changes shall be submitted to the GLAST Project CCB for approval.

VERSION	DATE	CHANGED BY	DESCRIPTION

OPEN ITEMS

This table lists the items that remain open as of the publication of this version of the document. As they are closed the resolutions will be incorporated into the body of the document.

ISSUE #	AFFECTED AREA	DESCRIPTION
1		LAT use of 64-bit integers and 64-bit floats. Will need to add a sets of codes for I12345678 and U12345678.

OPEN ACTIONS

This table lists the action items that remain open as of the publication of this version of the document. As they are closed the resolutions will be incorporated into the body of the document.

ACTION #	ACTIONEE	DESCRIPTION

TABLE OF CONTENTS

1.0 DATABASE OVERVIEW

In order to ensure that databases are exchanged in a compatible format and that all project elements communicate in a common language, this document has been designed to provide an overview of the GLAST database format. This document is based on the ITOS version 7.2 database and is modified to conform to the needs of the GLAST project.

The database is primarily composed of two parts: telemetry and command.

The telemetry component of the database contains a set of variables, referred to as "mnemonics". There are two broad classes of mnemonics: telemetry and non-telemetry mnemonics some of which are Mission Operations Center (MOC) reserved. Any mnemonic beginning with "GBL_" is reserved for the ground system software that will be used in the MOC. The values of telemetry mnemonics are sent from the spacecraft telemetry. The values of global mnemonics are set when the database is created and/or are set by the ground system, Spacecraft Test and Operation Language (STOL) procedures or configuration monitors (Configmons) running on the ground system.

The telemetry database also contains the maps required to extract the values of telemetry mnemonics from the telemetry stream, the limits on the mnemonic values, and the conversion data required to scale the value or map it to a text string.

The command part of the database contains spacecraft command specifications. Each command has a name, called the "command mnemonic", and consists of a set of fields. Each field is a contiguous set of bits in the telecommand packet.

Each field is named and may be associated with a set of named discrete values. In STOL, the term "submnemonic" is used to refer to a command field: It is either a field name, or a name associated with a discrete value from a command discrete value set.

1.1 References

The following documentation is the source material for this DFCD:

1. The ITOS 7.2 User's Guide, Database Overview - 2002/08/13
2. GLAST Project Mnemonic Naming Convention – (TBD)

2.0 DATABASE FILENAME and DIRECTORY STRUCTURE

All of the files used to define database exchange records must have unique names. Filenames will consist of all lowercase characters. To avoid filename conflicts, each filename must start with a pre-assigned prefix. ITOS database exchange files will reside in the multiple team directories starting from the $HOME/glast/dbx directory (i.e. $HOME/glast/dbx/sc, $HOME/glast/dbx/lat, etc.). Each team will locate their files in separate directories with the following name:

a. sc – Spacecraft
b. lat – LAT
c. gbm – GBM
d. ops – Operations

Each database exchange file filename must start with a pre-assigned prefix. Each database exchange file filename must have a '.dbx' suffix. It is also recommended that a version identifier be placed within the filename as well.

2.1 Database Exchange File Filename Examples

a. LAT tracker telemetry
   Filename: lat_tkr_tlm_v12.dbx
b. LAT tracker commands
   Filename: lat_tkr_cmd_v12.dbx
c. GBM data processing unit telemetry
   Filename: gbm_dpu_tlm_v34.dbx
d. GBM data processing unit commands
   Filename: gbm_dpu_cmd_v34.dbx
e. Spacecraft C&DH telemetry
   Filename: sc_cdh_tlm_v5.dbx
f. Spacecraft C&DH commands
   Filename: sc_cdh_cmd_v5.dbx

3.0 DATABASE EXCHANGE RECORD OVERVIEW

Database transactions will be in ASCII files containing information in the formats laid out below. The database software used for the GLAST ground system will be able to ingest files in this format.

Records are free format. The GLAST ASTRORT to ITOS conversion tool does not support multiple line record specifications. Therefore, there shall only be one record per line. The sequence '\nXXX,'—where '\n' is a new line character, 'XXX' is a Record Type, and ',' is a field separator—begins a new record. This means that there need not be a field separator between the end of one record and the start of the next as long as the new record begins at the start of a new line. Spaces or tabs preceding the record tag are allowed (and ignored).

Records may appear in any order subject the following limitations and with the following additional recommendations that will improve the human readability of the dbx files:

a. The vertical bar ("|") will be used as the standard field delimiter. Delimiters may be escaped with the backslash ('\') character and will not be interpreted as a delimiter if they appear inside double quoted strings.
b. Within each record type, fixed column widths will be used for each field to improve readability. Since blanks are not allowed, all records within a column should use the same number of characters.
c. Comments and blank lines will be used liberally to distinguish APID sets, commands with multiple "FLD" records, and configuration control information.
d. In telemetry definitions, the "TLM" and "PKT" records will be specified in the same byte order as the data is organized in the telemetry packet.
e. The "TLM", "PKT", "DSC", "ALG", and "LIM" records for a given telemetry mnemonic will all be contained in the same file unless the same "DSC", "ALG", or "LIM" records for that mnemonic are used by several subsystems.
f. In telecommand definitions, the "FLD" records for a given telecommand will immediately follow the "CMD" records for that telecommand.
g. "+" or "-" should NOT be used in actual mnemonics since it would pose a problem with the parser when used.

If there is no information for a particular field, or if the default value is desired, the field may be left blank. Fields missing from the end of a record are presumed blank.

All names and flags are case insensitive except for those within quoted strings. For record tags, mnemonic names, set names, and critical flags, upper and lower case characters are considered identical. Case is preserved only in description and DSC record state text fields. Names must begin with an alphabetic character and may contain only alphabetic and numeric characters and the underscore ('_'). The *GLAST Mnemonic Naming Convention* document contains the naming standard to be used. Names typically have length restrictions. The number of characters in names in transaction records is generally unlimited however some constraints do exist in order to keep software structures from getting to large. Also, the user should be aware that PDB files for the CMS and DTAS have fixed mnemonic name sizes of 16 characters. The following sizes apply:

Telemetry records:

Mnemonic names	- 16 characters.
Mnemonic units	- 64 characters.
Mnemonic string values	- unlimited.
Limit set names	- 255 characters.
Conversion set names	- 255 characters.
Discrete value strings	- 255 characters.
Maximum Packet size	- 65529 bytes.

Command records:

Mnemonic names	- 16 characters.
Field names	- 16 characters.
Submnemonic names	- 16 characters.
String Submnemonic value	- 16 characters.
Conditional critical exp	- 255 characters.
Checksum routine names	- 255 characters.
End item verification	- 255 characters.
Action routines	- 255 characters.

All records:

Short descriptions	- 64 characters.
Subsystem list	- 255 characters.
Subsystems per list	- 16.

Note that these records are called “transaction” records. There are no limitations on the numbers of each of these transaction records or the size of the database generated from them other than physical memory limitations. The database will exist and be maintained in one central location and this database will be considered the “master copy”. This master copy will be in the form an ASCII ITOS file called the ground system Project DataBase (PDB).

One of the main goals in the design of the record formats is to avoid dictating what database tools must be used. Therefore an ASCII file format that can be generated from common commercial software tools may be used to maintain parts of the database. Database changes can be made in the user’s preferred environment and produce a set of transaction records to apply against the master database, propagating those changes to the ground computer systems.

Not every element of the mission will require all of the functionality these records provide; but the format must be designed to allow the full range of options and operations of which the ground system is capable. The following topics are discussed in further details in the ensuing sections:

- Record Type
- keys

- numbers and other values
- dates and times
- type codes
- comments
- double quotes
- description fields

3.1 Record Type Field

The GLAST Ground System database software recognizes twelve types of transaction records, each of which is prefixed with a three-letter tag describing the contents of the record. The record types are:

'TLM'	a telemetry or global mnemonic definition
'ALG'	an analog conversion coefficient set definition
'DSC'	a discrete conversion state definition
'LIM'	a telemetry limit set definition
'PKT'	a telemetry packet item definition
'CMD'	a command mnemonic definition
'FLD'	a command field definition
'SUB'	a command field discrete value definition

The first field in each record contains one of these three-letter tags indicating the type of information contained in that record.

Note that these three-letter tags are keywords in the database exchange records and must be quoted, for example, if used in Field 3 of a DSC record.

3.2 Key Fields

The subsequent one or more fields contain the identifier or *key* for the record, naming the thing being defined. For example: a mnemonic name is the key for 'tlm' or 'cmd' records, and a command mnemonic name with a field name identifies a command field record.

3.3 Numbers and Other Values

3.3.1: Integer and Unsigned Values

Some fields (AppID in a PKT record, for example) require an integer value. Integer values may be specified in decimal, hexadecimal, or binary. (They may **not** be specified in octal).

The following, for example, all specify the same value:

```
17
017
0x11
0b010001
```

3.3.2 Floating Point Values

Other fields (the limit values in a LIM record, for example) require floating point values. The following are legal floating-point values:

```
3
3.1
.314e1
31.4e-1
-2.1415
```

3.4 Dates and Times

Date values represent a count of time from some epoch. The default epoch for the GLAST Project is 00:00:00 UTC on January 1, 2001 (The default MOC epoch is 00:00:00 on May 24, 1968). Any date telemetry mnemonic or command field may be associated with a different epoch, however, when it is defined in the database.

Alternate epochs are stored in telemetry mnemonics, and the epoch mnemonic is given in the 'field length' field of the PKT or FLD record. The actual field length is dictated by the type, and so need not be given for date or time types. The database build program, dbxodb, also provides an argument that allows you to specify an epoch mnemonic to be applied to all date and time telemetry mnemonics and command fields that do not otherwise have an epoch mnemonic defined for them.

Epoch mnemonics must be defined as dates, and should be set to the desired epoch. For example, for a spacecraft counting from the UNIX epoch, set an epoch mnemonic to 70-001-00:00:00. The mnemonic gbl_def_epoch is included in the DBX inputs as an example: It is set to the default MOC epoch.

The fine time field of the epoch mnemonic value gives the number of ticks per second. Each spacecraft tick is the equivalent of one microsecond.

3.4.1 LAT Time Format

LAT time format is copied directly from Spectrum Astro and it is two 32-bit words, big endian.

First word: Seconds from GLAST/LAT epoch
Second word: Microseconds within second.

GLAST/LAT epoch (also Spectrum Astro definition):

2001-01-01 00:00:01 minus one second

3.4.2 GBM Time Format

The GBM time format is as follows:

coarse time: 4-byte unsigned integer, units are 0.1 second, so rollover
occurs ~13.61 years after epoch zero

fine time: 2-byte unsigned integer, units are 2 microseconds, hardware
causes rollover at 50000 (=0.1 second)

3.5 Type Codes

These are the possible values for

- A TLM record's field 6, the "destination type".
- A PKT record's field 7, the "source type".
- A FLD record's field 5, the "destination type".

Types are:

One byte integral types	U1 I1	These hold signed (I1) or unsigned (U1) numbers that are no more than 8 bits wide and that do not span octets. Signed numbers use two's complement encoding. If width is otherwise unspecified in the transaction records, a width of 8 is assumed. These may have analog or discrete conversions.
Two byte integral types	U12 I12	These hold signed (I12 and I21) or unsigned (U12 and U21) numbers that are no more than 16 bits wide and that span two

	I12 U21 I21	numbers that are no more than 16 bits wide and that span two octets. (8-bit or smaller numbers that are contained within one octet could be U1 or S1). Two octets get transmitted in big-endian order (high order byte first) for U12 and S12 and little-endian order for U21 and S21. Signed numbers use two's complement encoding. If width is otherwise unspecified in the transaction records, a width of 16 is assumed. These may have analog or discrete conversions.
Four byte integral types	U1234 I1234 U4321 I4321 U3412 I3412 U2143 I2143	These hold signed (I1234, I4321, I3412, and I2143) or unsigned (U1234, U4321, U3412, and U2143) numbers that are up to 32 bits wide and that span at least three octets (values that span only one or two octets could be one of the smaller integral types). Four octets get transmitted: for U1234, I1234, the octets get transmitted in big-endian order (high order byte first); for U4321 and I4321 the octets get transmitted low order byte first; for U3412 and I3412 the octets get transmitted in the bizarre order next-to-low order byte first, low order byte second, high order byte third, and next-to-high order byte last, and for U3412 and I3412 the octets get transmitted in the equally bizarre order next-to-high order byte first, high order byte second, low order byte third, and next-to-low order byte last. Signed numbers use two's complement encoding. If width is otherwise unspecified in the transaction records, a Not Supported by AstroRT.
Four byte floating point	F1234 F3412 F4321 F2143	32-bit IEEE-754 floating point numbers. Four octets get transmitted: for F1234 the octets get transmitted in big-endian order (high order byte first); for F4321 the octets get transmitted low order byte first; for F3412 the octets get

		low order byte third, and next-to-low order byte last. The only legal width is 32. These may have analog or discrete conversions.
Eight byte floating point	F12345678 F78563412 F87654321 F43218765 F21436587	64-bit IEEE-754 floating point numbers. Eight octets get transmitted: for F12345678 the octets get transmitted in big-endian order; for F78563412 the next-to-low order byte is transmitted first and the next-to-high order byte last; for F87654321 the octets get transmitted in strict little-endian order; for F43218765 the octets get transmitted in a kind of ‘little-endian with long-words swapped’ order; and for F21436587 the next-to-high order byte is transmitted first and the next-to-low order byte last. If width is otherwise unspecified in the transaction records, a width of 64 is assumed. (In fact, the only legal width is 64). These may have analog or discrete conversions.
CUC time		
	RTIME12	A 3-octet relative time. The first octet (or byte) is the number of seconds, the second and third octets form a 16 bit little-endian number (I.e. the third byte is the high order byte) that represents the number of fractional (2^-16) seconds.
	RTIME20	A 2-octet relative time. The octets are transmitted in little-endian order; the two octets form a 16-bit number of seconds.
	RTIME40 RTIME42	A 4-octet (RTIME40) or 6-octet (RTIME42) absolute time. The first four octets form a 32 bit little-endian (i.e. I4321) number of seconds that represents the moment in time that many seconds after the epoch. For RTIME42, the last two octets format a 16 bit little-endian number that represents the
	TIME12	A 3-octet relative time. The octets are transmitted in big-endian order; the first octet (or byte) is the number of seconds, the second and third octets form a 16-bit number that represents

	TIME42	octets are transmitted in big-endian order; the [first] four octets form a 32 bit number of seconds that represents the moment in Time that many seconds after the epoch. For TIME42 the last two octets format a 16-bit number that represents the number of fractional (2^-16) seconds.
Non-CUC time		
	RTIMET42	A 6-octet absolute time. The first four octets form a 32 bit little-endian (i.e. I4321) number of 1/10 seconds that represents the moment in time that many seconds after the epoch. The last two octets format a 16 bit little-endian number that represents the number of fractional (2^-16) seconds. Rollover of the date will occur about 6.8 years past the epoch.
	TIMET42	A 6-octet absolute time. The first four octets form a 32 bit big-endian (i.e. I1234) number of 1/10 seconds that represents the moment in time that many seconds after the epoch. The last two octets format a 16 bit big-endian number that represents the number of 2-microsecond (2^-16) intervals. Rollover of the date will occur about 6.8 years past the epoch. This is used in conjunction with GBM time, which is in tenths of seconds.
	RTIME44	An 8-octet absolute time. The first four octets form a 32 bit little-endian (i.e. I4321) number of seconds that represents the moment in time that many seconds after the epoch. The last four octets format a 32 bit little-endian number that represents the number of fractional seconds up to 2^-32. The default fractional tick if not specified in the epoch mnemonic is (1e-6) one microsecond.
	TIME 44	An 8-octet absolute time. The first four octets form a 32 bit big-endian (i.e. I1234) number of seconds that represents the moment in time that many seconds after the epoch. The last four octets format a 32 bit big-endian number that represents the number of fractional seconds up to 2^-32. The default fractional tick if not specified in the epoch mnemonic is (1e-6) one microsecond. This is used in conjunction with LAT time.

3.6 Comments

Comments begin with a '#' character and continue to the end of the line. Quoted \# is a quoted '#' that does not begin a comment and provides the means to get '#' in the description field. This may also be done by simply quoting the description filed. When a comment occurs in the middle of a record, the comment is treated as white space. For example, the following is allowed:

```
ALG|CNVSET1|+|                    # Analog conversion CNVSET1:
          | -3.6065400e+01  # C[0]
          | 1.78768000e-01  # + C[1] X
          | -5.9817700e-04  # + C[2] X^2
```

3.7 Quoted Values

The information provided for any field may appear inside double quotes. Delimiter and comment characters are not interpreted when they appear in a quoted string.

3.8 Description Fields

Most records contain a "description" field, which may contain either or both of the item's long description and the item's short description. If the description field contains the string '<HTML>', everything before '<HTML>' is the short description and '<HTML>' begins the long description. Otherwise, there is no long description. Short descriptions may get truncated if longer than 62 characters. Short descriptions are used in event messages, interactive prompts, and printed reports. The <HTML> field may be ignored during ingest into the AstroRT system since it is a description field.

As implied by '<HTML>', long descriptions are written in Hypertext Markup Language (HTML) and are intended to be read using a Word Wide Web (WWW) browser. Long descriptions may include pictures and may have hypertext links to other descriptions or documents.

The following shows a transaction record with both short and long descriptions:

```
TLM,GBL_PROCPATH,+,,ITOS,CHAR,999,,,,,,".",T,
   "List of directories STOL searches when looking for procs.
    <html>The colon-separated list of directories, modeled after the
    Unix environment variable <var>PATH</var>, that
    <font size=-1>STOL</font>
    searches when looking for a proc.
    <p>
    If the proc occurs in more than one of the directories in
    <var>GBL_PROCPATH</var>, the proc from the first of those
    directories is used.
```

<p>
See also <a href=\"GBL_PAGEPATH.html\">GBL_PATHPATH</a>."

3.8.1 Using relative URLs to reference other descriptions

Long descriptions allow items reference each other using relative URLs. Long descriptions are inserted into machine-generate html files, which are organized as follows:

htmldir/dir.html	Machine generated table of contents.
htmldir/packets	Directory containing machine generated descriptions of the packet maps. Files are named using '"app%04d.html",appid' where 'appid' is the packet's AppID number: app0001.html contains the machine-generated description of the appid 1 packet. Long descriptions from PKT records may be inserted in these files.
htmldir/mnemonics	Directory containing machine generated descriptions of telemetry mnemonics. Files are named using '"%s.html",mnemonic' where 'mnemonic' is the uppercase mnemonic name: SCCMDCNT.html contains the machine-generated description of mnemonic SCCMDCNT. Long descriptions from TLM, LIM, ALG, and DSC records may be inserted in these files.
htmldir/commands	Directory containing machine generated descriptions of commands. Files are named using '"%s.html",command' where 'command' is the uppercase name of the command: SNOOP.html contains the machine-generated description of command SNOOP. Note that command names are uppercase! Long descriptions from CMD, FLD, and SUB records may be inserted in these files.
htmldir/subsystems	Directory containing machine generated descriptions of subsystems. Files are named using '"%s.html",subsystem' where 'subsystem' is the uppercase subsystem name: ACS.html contains the machine-generated description of the ACS subsystem. Long descriptions from SSI records are inserted in these files.

---

[1] This may change! The side effect of allowing more than one record per line is that 'del', 'ssi', 'tlm' , etc. must be quoted whenever they're not beginning a new record, for example:
dsc,transaction,add,+,1
dsc,transaction,mod,+,1
dsc,transaction,"del",+,1

Chapter 4.0 TELEMETRY RECORDS

The TLM records define where a value extracted from a packet is stored in MOC's real-time value tables. One TLM record can be referenced in multiple PKT records because a data item can come down in a stream multiple times but it is only stored in one place. The database compiler ‘DBXODB’ will warn of the use the same TLM record in multiple PKT records because it is an unusual case so that the user is sure it was not done by mistake.

Along with the TLM mnemonic record, optional limit and conversion records that are specified in the PKT record must be defined. It is not necessary to define unique limit and conversion records for each PKT because they can be shared and reused in any PKT record for any AppID.

LIMITS are defined by using the LIM record. Limit records define upper and lower bounds of a number to determine a YELLOW and RED state or limit. These limits will trigger events messages when the thresholds are crossed in two consecutive packets. Conversion of the PKT data is defined by two types, ANALOG which use the ALG records and DISCRETE which use the DSC records. An ANALOG conversion specifies how to convert analog value to engineering units by applying a polynomial equation. A DISCRETE conversion specifies how to take a numeric value and convert it to a text string.

In telemetry definitions, the “TLM” and “PKT” records will be specified in the same byte order as the data is organized in the telemetry packet.

The “TLM”, “PKT”, “DSC”, “ALG”, and “LIM” records for a given telemetry mnemonic will all be contained in the same file unless the same “DSC”, “ALG”, or “LIM” records for that mnemonic are used by several subsystems.

The following is a telemetry mnemonic definition for a system variable. It is included here because it contains a nice bit of HTML documentation. This mnemonic doesn't have an associated PKT record because it doesn't come down in telemetry. You might create something like this to hold pseudo-telemetry produced by the configuration monitor.

```
TLM |GBL_CM_TXPORT |+||ITOS_CMD |S1|32|||1|||CLIENT_TCP|F|
   "Command destination host connection type.
   <html>Determines whether ITOS initiates a TCP/IP connection,
   accepts a TCP/IP connection, or makes a UDP/IP association for
   spacecraft commanding.  For security reasons, ground stations may
   want to initiate the TCP connection.
   <p>Values are:
   <dl>
   <dt><code>client_tcp</code>
   <dd> ITOS initiates a connection from an arbitrary port on the
       localhost to port <a href=\"GBL_CMD_PORT.html\">
       GBL_CMD_PORT</a> on host <a href=\"GBL_CMD_HOST.html\">
       GBL_CMD_HOST</a>.
   <dt><code>server_tcp</code>
   <dd> ITOS listens for a connection from any host and port on port
GBL_CMD_PORT.
   <dt><code>udp</code>
   <dd> ITOS makes a UDP association using the same host and port
       configuration as the <code>client_tcp</code> mode.  If
       GBL_CMD_HOST resolves to a multicast (class D) address, the
       multicast time-to-live is set automatically to 127.
   <dt><code>serial</code>
   <dd> ITOS opens the serial port (RS-423/RS-232) given by <a
   href=\"GBL_CMD_FILE.html\">GBL_CMD_FILE</a> and configures it
       according to <a
       href=\"GBL_CMD_SERIAL.html\">GBL_CMD_SERIAL</a>.
   <dt><code>file</code>
   <dd> ITOS sends commands to the file given by <a
       href=\"GBL_CMD_FILE.html\">GBL_CMD_FILE</a>.
   </dl>"
```

The telemetry records are

'TLM'	define a telemetry mnemonic,
'ALG'	define an analog conversion,
'DSC'	define a discrete conversion,
'LIM'	define a limit set,

‘PKT’ define an item in a packet map, and

4.1 TLM Record

The ‘TLM’ record defines telemetry and global mnemonics. Note that because mnemonics need not be in the telemetry stream, the definition of a mnemonic does *not* include where in the telemetry stream that mnemonic may be found. Non-telemetry mnemonics may be system state variables (a so-called “global mnemonic” that begins with “GBL_”) or a user-defined mnemonic created to hold a derived value or a constant value. Mnemonics also may appear multiple times in the telemetry stream, in the same packet or in different packets.

In telemetry definitions, the “TLM” and “PKT” records will be specified in the same byte order as the data is organized in the telemetry packet.

Note: The “TLM”, “PKT”, “DSC”, “ALG”, and “LIM” records for a given telemetry mnemonic will all be contained in the same file unless the same “DSC”, “ALG”, or “LIM” records for that mnemonic are used by several subsystems.

Field 1	‘TLM’, indicating the record type.
Field 2	The telemetry “mnemonic” is the name of the telemetry data point or ground system global value. The value is accessed through this name, just like a variable in a program (which it essentially is in the case of STOL procedures). Mnemonic names must begin with a letter, may contain only alphanumeric characters or underscore (‘_’), and may be up to 16 characters long. Mnemonics that begin with “GBL_” are reserved for system use and the user should refrain from using them. Reference *GLAST Project Mnemonic Naming Convention* document for mnemonic naming conventions.
Field 3	The operation symbol ‘+’, although not supported by AstroRT, is still required by ITOS.
Field 4	Mnemonic ID. This is a historical artifact from the FAST mission and is normally left blank. For FAST, this was a unique integral number that identified the mnemonic between 0 and 65534. A blank ID defaults to 65535.
Field 5	List of subsystem names (see ssi) indicating which subsystems this mnemonic is associated with. Up to 16 subsystem names may be specified using spaces to separate the names.
Field	The “destination type” of the mnemonic. Indicates how the mnemonic’s value is

6	stored and used within the MOC/ground system. Also used as the default "source type" if none is specified in the PKT record.
Field 7	The size of the mnemonic's value, in bits or, if the "type" is a string type, in bytes. If blank, this field value will be inferred from the type, if possible. For example, a U12 defaults to 16 bits. Signed and unsigned integers may not be larger than 32 bits; floating-point values may be 32 or 64 bits; CUC times and dates may be 16, 24, 32, or 48 bits; BCD values must be 64 bits; and PB5 times are 48 or 56 bits; and strings have no default size. Note that the size given here is used to limit the range of values that STOL may assign to a mnemonic. For example, a size of 2 bits limits an unsigned integer mnemonic's values to the range 0 to 3.
Field 8	Engineering units string (up to 64 chars). Used when displaying the mnemonic's value.
Field 9	Not supported by AstroRT.
Field 10	Array length (default = 1). Reserved for future use.
Field 11	Delta Limit. This field may contain a name, a number, or a name and a number separated by white space. The delta limit is the maximum amount the value may change between samples. If the field contains a name, that name is the limit definition (see lim) that specifies the red/yellow limits for this mnemonic. (If the field does not contain a name, the mnemonic has no red/yellow limits).
Field 12	The name of the analog (see alg) or discrete (see dsc) conversion to apply to this mnemonic value when displaying the value or if it is referenced through the STOL 'p@' operator. If blank, the mnemonic has no conversion.
Field 13	AstroRT does not have functionality to initialize the telemetry Current Value Table when a database is loaded.
Field 14	'T' or 'F'. The default value for observatory telemetry is 'T'. The default value for ground telemetry will be 'F'. This will allow ground directives to alter the telemetry state of ground mnemonics, but not the states of observatory mnemonics.
Field 15	The mnemonic description.

4.2 ALG Record

Analog conversions, defined in ALG records, are intended for converting an integer number of "counts" (the output of an analog to digital converter, for example) into a floating-point value in "engineering units", such as volts, amps, degrees, etc. The ALG record defines the coefficients for an 8th order polynomial. The integer or floating-point telemetry value is applied to the polynomial and the result is a floating-point value.

Note: The “TLM”, “PKT”, “DSC”, “ALG”, and “LIM” records for a given telemetry mnemonic will all be contained in the same file unless the same “DSC”, “ALG”, or “LIM” records for that mnemonic are used by several subsystems.

Field 1	‘ALG’, indicating the record type.
Field 2	The analog conversion definition name, which is used in the ‘tlm’ record’s conversion definition name. Conversion names must begin with a letter may contain only alphanumeric characters or underscore (‘_’), and may be up to 255 characters long. Analog and discrete conversions may not have the same name.
Field 3	The operation symbol ‘+’, although not supported by AstroRT, is still required by ITOS.
Field 4	A floating point value representing C0 in the polynomial C0 + C1*X + C2*X^2 + ... + C7*X^7 The default is ‘0.0’.
Field 5	A floating point value representing C1, the coefficient of X^1. The default is ‘0.0’.
Field 6	A floating point value representing C2, the coefficient of X^2. The default is ‘0.0’.
Field 7	A floating point value representing C3, the coefficient of X^3. The default is ‘0.0’.
Field 8	A floating point value representing C4, the coefficient of X^4. The default is ‘0.0’.
Field 9	A floating point value representing C5, the coefficient of X^5. The default is ‘0.0’.
Field 10	A floating point value representing C6, the coefficient of X^6. The default is ‘0.0’.
Field 11	A floating point value representing C7, the coefficient of X^7. The default is ‘0.0’.
Field 12	The analog conversion’s description.

4.3 DSC Record

Discrete conversions, defined in DSC records, transform a range of numeric values into a set of text strings. The telemetry value is compared to each range in the set. If the value falls within a range, the state text associated with that range is displayed. There shall be no overlapping ranges. Since AstroRT ignores the high end value of the field and ITOS uses both the high and low end values, . the user must populate the low and high range with identical values. The DSC records will then have a single state and not a range for each value.

Note: The “TLM”, “PKT”, “DSC”, “ALG”, and “LIM” records for a given telemetry mnemonic will all be contained in the same file unless the same “DSC”, “ALG”, or “LIM” records for that mnemonic are used by several subsystems.

Field 1	‘DSC’, indicating the record type.
Field 2	The discrete conversion definition name, which is used in the ‘tlm’ record’s conversion definition name (field 12). Conversion names must begin with a letter may contain only alphanumeric characters or underscore (‘_’), and may be up to 255 characters long. Discrete and analog conversions may not have the same name.
Field 3	The state text. This is the string displayed if the mnemonic value falls in the given range. This string may contain non-alphanumeric characters (in which case it should be quoted). There is no max length of any one state text but however the strings are truncated after 255 characters. .
Field 4	The operation symbol ‘+’, although not supported by AstroRT, is still required by ITOS.
Field 5	A floating point value representing the low value for the range. The default is ‘-DBL_MAX’.
Field 6	A floating point value representing the high value for the range. The default is ‘DBL_MAX’. Values are compared with the discrete range as: low <= value <= high
Field 7	Not supported by AstroRT..
Field 8	Not supported by AstroRT.
Field 9	The discrete conversion’s description.

4.4 LIM Record

The LIM record defines limits for an integer or floating-point telemetry mnemonic. A limit set consists of two concentric ranges called the “yellow limits” (fields 5 & 6) and “red limits” (fields 4 & 7). A limit definition may contain more than one limit set. The system chooses which limit set to apply to a mnemonic using the “limit switch”, explained below.

Note: The “TLM”, “PKT”, “DSC”, “ALG”, and “LIM” records for a given telemetry mnemonic will all be contained in the same file unless the same “DSC”, “ALG”, or “LIM” records for that mnemonic are used by several subsystems.

Field 1	‘LIM’, indicating the record type.
Field 2	The limit definition name, which is used in the ‘tlm’ record’s limit field (field 11). Limit definition names must begin with a letter, may contain only alphanumeric characters or underscore (‘_’), and may be up to 255 characters long.
Field 3	The operation symbol ‘+’, although not supported by AstroRT, is still required by ITOS.
Field 4	A floating point value representing the red low limit. The default is ‘-DBL_MAX’.
Field 5	A floating point value representing the yellow low limit. The default is ‘-DBL_MAX’.
Field 6	A floating point value representing the yellow high limit. The default is ‘DBL_MAX’.
Field 7	A floating point value representing the red high limit. The default is ‘DBL_MAX’.
Field 8	The name of the limit switch mnemonic. The default is blank, indicating no limit switch is used. The limit switch is discussed below.
Field 9	A floating point value representing the limit switch low value. The default is ‘-DBL_MAX’.
Field 10	A floating point value representing the limit switch high value. The default is ‘DBL_MAX’.
Field 11	The inversion flag; ‘T’ or ‘F’. Discussed below; the default value is ‘F’.
Field 12	The limit set description.

Limit definitions allow the ground system to support two limit sets: red and yellow.

The limit switch (fields 8 - 10) is used to determine which set is to be used for limit checking. The raw (unconverted) value for the given “limit switch mnemonic” (field 8) (the name of any integer or floating-point mnemonic) is tested against the range presented by the limit switch low (field 9) and high (field 10) values as:

```
if (fld9_low != fld10_high) {
   if (fld9_low <= fld_8_mnemonic_value < fld10_high) {
        Use this limit set;
    } else {
        Keep on looking;
      }
    } else { // fld9_low == fld10_high
      if (fld9_low == fld_8_mnemonic_value) {
        Use this limit set;
    } else {
        Keep on looking;
      }
    }
```

A mnemonic must be received in the same state of violation or in-limits twice consecutively before the ground system reports the limit condition.

4.6 PKT Record

A telemetry mnemonic's value may be extracted from any telemetry packet. It may be present multiple times in a packet and in multiple packets. Each PKT record defines how to extract a single telemetry value from single occurrence in one packet.

In telemetry definitions, the "TLM" and "PKT" records will be specified in the same byte order as the data is organized in the telemetry packet.

Note: The "TLM", "PKT", "DSC", "ALG", and "LIM" records for a given telemetry mnemonic will all be contained in the same file unless the same "DSC", "ALG", or "LIM" records for that mnemonic are used by several subsystems.

Field 1	'PKT', indicating the record type.
Field 2	The packet application ID. An integer value. Legal values are 0 to 65535 however CCSDS packets are limited from 0 to 2047.
Field 3	A telemetry mnemonic name, as specified in Field 2 of a TLM record. Identifies the TLM mnemonic this record's data point unpacks into.
Field 4	The array index, an integer value used for mnemonic arrays the elements of which are not telemetered at constant intervals in a packet or are not in the same packet. Normally, array elements are in one packet at a constant distance from one another. The "offset between array elements" (field 11) gives this constant distance. Reserved for future use.
Field 5	The operation symbol '+', although not supported by AstroRT, is still required by ITOS.
Field 6	Unused.
Field	The "source type" of the mnemonic (see type codes). Indicates how the

7	mnemonic’s value is unpacked. If not specified, a default is derived from the mnemonic’s TLM record’s “destination type” (field 6).
Field 8	An integer value representing the “starting byte” where the data value begins, measured from the beginning of the packet primary header. In other words, the first byte in the packet primary header is byte 0. The starting byte plus the length of the packet item in bytes must not exceed the maximum packet length of 65529.
Field 9	An integer value representing the “starting bit” within the starting byte where the data begins. The most significant bit is bit 0. If blank, this will be assumed to be zero.
Field 10	The “field length”, an integer value representing the number of bits comprising the data. If “source type” (field 7) is CHAR or S then length is in bytes. If blank, the length is inferred from the type, or is copied from the “size” (field 7) element of the mnemonic’s TLM record. If source type is a TIME or DATE type then this field can be used to identify a telemetry mnemonic that defines the epoch used to decom the data. If omitted the default epoch is used. GBL_DEF_EPOCH can be used or any other DATE type mnemonic defined. The subseconds define how to convert from spacecraft subseconds to microseconds for both DATE and TIME mnemonics. Note that the end bit given by (“start bit” + “length” - 1) may not fall outside the field selected by the source type. It is important to remember this rule when specifying a map for unpacking arrays.
Field 11	The “array offset”, an integer value representing the offset in bits between array elements. Not used unless the mnemonic is an array. The default value comes from field 10 (the field length). If 0 only one array element is extracted.
Field 12	The selector definition name. Currently unused, this is intended to allow a reference to a set of SEL records.
Field 13	description.

5.0 COMMAND RECORDS

The CMD records define command mnemonics. Each command mnemonic maps to a particular command packet. CCSDS commands have an AppID; commands may also have a function code; these are imbedded in the telecommand packet primary and secondary headers, respectively. Furthermore, each command is composed of a set of named data fields, defined in FLD records. Each FLD record defines a set of contiguous bits in the command packet data field.

FLD records may contain a high-low range restricting the values to which the field may be set. Each field also may have associated with it a set of discrete values to which the field may be set. Defined in one or more SUB records, this set of values is the inverse of a telemetry mnemonic's discrete conversion: They provide a set of names each of which map to a fixed value for the field.

In order to understand the command database, it is necessary to understand how spacecraft commands are specified in the STOL language. The STOL syntax for a spacecraft command is:

```
        [”cmd “ | /]<mnemonic> <submnemonic>[, <submnemonic>, ...]
```

for example:

```
        /heaterctl shade, temp=22.4
        /heateroff shade
```

The '/' indicates the following word is a spacecraft command. The second command could have been given alternatively, for example, as 'cmd heateroff shade'. 'heaterctl' and 'heateroff' are command mnemonics. 'shade' and 'temp' are submnemonics; that is, parameters to the 'heaterctl' or 'heateroff' commands.

Notice the two forms of submnemonic presented. 'shade' is often referred to as a "fixed" submnemonic, and 'temp' is called a "variable" submnemonic. In the database, 'temp' is the name of a command data field (FLD), and 'shade' is one of the names from a set of discrete values associated with another field.

Note that it would have been possible to enter the command as:

```
        /heateroff heater=shade
```

assuming that 'heater' is the name of a field referencing a set of fixed values including one named 'shade'. This syntax is required if the all the discrete value names don't map to a unique field for the given command. For example, for the command:

```
        /set_relays on, off, on, on
```

it is not possible to tell which "on" or "off" refers to which field. This requires a syntax like:

```
        /set_relays main=on, aux=off, inst=on, acs=on
```

The field names for each command must be unique for that command. Two or more commands can have fields with the same name, but no two fields within a single command can have the same name.

If a discrete value name is unique among all discrete value sets referenced by all fields in a single command, and among the field names for the command, then that name may be given alone in a STOL spacecraft command to specify both the field and the value to which that field is being assigned. (Like "shade" in the examples.) Otherwise, the syntax '<field name> = <discrete value name>' must be used.

If a field definition contains a value range or does not reference a discrete value set, then the field value may always be specified to STOL in the “variable submnemonic” syntax. (Like ‘temp’ in the examples.)

If a field definition references a discrete value set but does not contain a value range, then the field value must be set using one of the discrete value names. There will be a STOL mode to override this restriction for testing purposes.

Normally, STOL requires that a value be given explicitly for all fields. However, it is possible to set up the database so that commands may contain “hidden” or “optional” fields.

If an FLD record references a set of discrete values, and one value in the set is mapped to the name “default”, then the “default” value is automatically used for the field if no other value is given for it in the STOL command. This is then an optional field, from the STOL perspective.

If a FLD record specifies a range where the low and high values are equal, so that the field can only take one value, then that value automatically will be supplied; there’s no need to create a SUB record with the name “default” and the STOL user doesn’t have to specify the [one and only] value for the field.

Comments and blank lines will be used liberally to distinguish AppID sets, commands with multiple “FLD” records, and configuration control information.

In telecommand definitions, the “FLD” records for a given telecommand will immediately follow the “CMD” records for that telecommand.

The following is an example database excerpt for the two commands “heaterctl” and “heateroff” used as examples above. Note that “heateroff” has the same command AppID and function code as “heaterctl”, but the “temp” is set to a fixed, “hidden”, value. These command might be defined as follows:

```
SSI,thermal,+,”Heaters and stuff”
CMD,heaterctl,+,1,1,thermal,,,80,,,,,,Command to control various heaters
FLD,heaterctl,heater,+,U1,,8,0,8,,,,heaters,Select heater to control
FLD,heaterctl,temp,+,F12345678,,9,0,64,,-10,98.6,,
                Temperature to which heater should be set
SUB,heaters,shade,+,1,,Instrument shade
SUB,heaters,body,+,2,,Instrument body
SUB,heaters,detect,+,4,,Detector
SUB,heaters,all,+,7,,All heaters
CMD,heateroff,+,1,1,thermal,,,80,,,,,,Command to control various heaters
FLD,heateroff,heater,+,U1,,8,0,8,,,,heaters,Select heater to control
```

```
FLD,heateroff,temp,+,F12345678,,9,0,64,,-10.0,-10.0,noheat,
                               Temperature to which heater should be set
SUB,noheat,default,+,-10.0,,Temp setting to turn heaters off
```

The command records are

'CMD'	define a command,
'FLD'	define a command field, and
'SUB'	define a discrete value for a command field.

5.1 CMD

Field 1	'CMD', indicating the record type.
Field 2	The "command mnemonic" is the name of the command and is how STOL procedures reference the command. Command mnemonic names must begin with a letter, may contain only alphanumeric characters or underscore ('_'), and may be up to 16 characters long. Reference *GLAST Project Mnemonic Naming Convention* document for mnemonic naming conventions.
Field 3	The operation symbol '+', although not supported by AstroRT, is still required by ITOS.
Field 4	This field is ignored for non-CCSDS commands.

The "application ID" and "default virtual channel" are combined in this field; values for this field are either integer value or two integer numbers separated by 'VC'.

In the first form, where the field contains only one number, that number is the value for the CCSDS telecommand packet's primary header application ID field. Valid values are 0 to 2047. The default virtual channel is said to be unspecified.

In the second from, the first number is the application ID and the second number is the default virtual channel in the range of 0 to 63. Reserved for future use:

The default virtual channel is used in command processing: unless overridden in STOL's CMD directive, the command will get transmitted on the default virtual channel. If the default virtual channel is unspecified, the command is transmitted on the virtual channel identified by GBL_VC.

It is expected that most commands will leave the default virtual channel unspecified.

Field 5 The “function code” or command type indicator.

This is either an 15-bit unsigned integer value, which both indicates that this is a command and specifies the value for the CCSDS telecommand packet secondary header “function code”.

Or this is one of the strings:

CCSDS	This is a CCSDS command (without secondary header).
RAW	This is a raw, non-CCSDS command.

Field 6 List of subsystem names (see ssi) indicating which subsystems this command is associated with. Up to 16 subsystem names may be specified using spaces to separate the names.

Field 7 Not used by conversion tool.

Field 8 Not used by conversion tool.

Field 9 The “command length in bits”. (“In bits” rather than “in bytes” is a historical artifact; the length in bits is always a multiple of eight!). In most cases, you should leave this field blank, in which case dbxodb will supply the minimum command length that will hold all of the fields specified for the command.

For CCSDS commands this is the value of the primary packet header’s “packet length” field multiplied by 8; that is, this is the length in bits of the application data field including any secondary header, minus 8 bits.

For example, consider a command packet that is 15 bytes total (including **all** headers): Since the packet primary header is 6 bytes long, the application data field (plus secondary header) is 9 bytes long. So the “command length in bits” is 9 bytes times 8 bits per byte minus 8 bits, or 64 bits.

For non-CCSDS commands, this is the length of the whole command packets. So the “command length in bits” of a 15-byte non-CCSDS command is 15 bytes times 8 bits per byte, or 120.

The length entered here should **not** include the length of any checksum specified in this command record or through GBL_CHKSMROUT.

Field 10 Not used by conversion tool.

Field 11 The “critical flag” indicates that the ground system should prompt for permission to send the command whenever the command is issued. This field may be set to one of:

H	hazardous
R	critical
C	conditionally critical
N	not critical (default)

The MOC software does not distinguish between “hazardous”, “critical” and “conditionally critical”. This is a filed that will be used operationally, but is not recognized by AstroRT.

Field 12	Not used.
Field 13	Not used by conversion tool.
Field 14	Not used.
Field 15	description.

5.2 FLD

In telecommand definitions, the “FLD” records will immediately follow the “CMD” records for a given telecommand.

Field 1	‘FLD’, indicating the record type.
Field 2	Command mnemonic names must begin with a letter, may contain only alphanumeric characters or underscore (‘_’), and may be up to 16 characters long.
Field 3	field name (up to 16 chars). The “field name” is a descriptive name for the field.
Field 4	The operation symbol ‘+’, although not supported by AstroRT, is still required by ITOS.
Field 5	The “destination data type” is analogous to the “source data type” in the PKT record. It is the field’s data type, which indicates the type of value for the field (unsigned integer, IEEE floating point. string, date, etc) and byte order of the value.
Field 6	The “array size” gives the array length if this record defines an array field. Reserved for future use.

Field 7 The “starting byte” is the byte where the field begins; byte 0 is the first byte of the command packet. I.e., for CCSDS commands, byte 0 is the first byte of the packet primary header. Except for *GBL_LCLHDR* special commands, the starting byte of a command field for MUST be greater than 5 for CCSDS commands or it will be flagged as an error.

Field 8 The “starting bit” is the bit within the destination field where the data field begins. Bits are numbered big endian starting with 0; the high order bit of each byte is bit 0. Note that specifying a non-zero starting bit only makes sense for signed or unsigned integral values. If blank, this will be assumed to be zero.

Field 9 The “field length”.

For signed or unsigned integral data types, this is the length in bits and must be less than or equal to the maximum length for the type. If not specified, the default is the maximum length for the type, i.e. 8 for UI, 16 for U12, etc.

For TIME or DATE types, this field can be used to identify a telemetry mnemonic that defines the epoch used to encode the data. If omitted the default epoch is used. GBL_DEF_EPOCH can be used or any other DATE type mnemonic defined. The subseconds define how to convert from microseconds to spacecraft subseconds for both DATE and TIME fields.

For all other data types, this field should be left blank.

Note that for arrays, this is the length of each element, not of the whole array.

Field 10 Not used by conversion tool.

Field 11 field value range lower bound

Field 12 field value range upper bound

The “field value range lower” and “upper” bounds, provide a range of valid values for the field. The field value must satisfy the expression:

lower_bound <= field_value <= upper_bound

Note that field values are automatically bounded somewhat by data type and field length, such that a 2-bit unsigned field is automatically constrained to be valued from 0 to 3, for example. So, if no range is specified, the value will be checked only to see that it fits in the given field length. If no SUB records are defined for this field and lower and upper bounds are the same value then that value will act the same as the “default”. If so, STOL will not allow a value for this field making it invisible or hidden.

Field 13 The “discrete value set name” gives the name of a discrete value set containing a set of values to which the field may be set. As noted in the commentary at the head of this section, this field and the preceding two fields (the value range) interact to

	determine how the field value may be set.
Field 14	description.

5.3 SUB

Field 1	'SUB', indicating the record type.
Field 2	The "discrete value set name" is the name used in field 13 of the FLD record. It names a set of SUB records. This may be up to 16 characters long.
Field 3	The "value name" is the name for this discrete value in the set. This is the name used in STOL when sending the command to specify the associated value for some field. This may be up to 16 characters long. The special name "default" marks this as the field's default value, i.e. the value the field will be set to if no other value is specified.
Field 4	The operation symbol '+', although not supported by AstroRT, is still required by ITOS.
Field 5	The "fixed value" is the fixed value associated with "value name". If this value is associated with a FLD record of string type (S1 or S21) the maximum string length of the fixed value is 16 characters.
Field 6	Not used by conversion tool.
Field 7	description.

6.0 SPECIAL COMMANDS AND MNEMONICS

6.1 Special Mnemonics

The following mnemonics must be defined. If missing dbxodb will create them with specific initial values!

GBL_MISSION	TLM,GBL_MISSION,+,,,S,32,,,,,,”trace”,, “The mission this database is for.”
GBL_DBVERS	TLM,GBL_DBVERS ,+,,,S,32,,,,,,”4.1”,, “Database version.”
GBL_DEF_EPOCH	TLM,GBL_DEF_EPOCH,+,,ITOS_DB,TIME,,,,1,,,68-145-0:0:0.065536,F, “Default EPOCH time adjustment. GBL_DEF_EPOCH is the date of the spacecraft epoch. The default initial value is ‘Midnight May 24,1968’ the MOC epoch. The factional part is the conversion denominator to convert from spacecraft subseconds to microseconds, 065535 is default for the MOC. The leading zero is needed to account for a 6-digit number. MOC Initial Value: ‘68-145-0:0:0.065536’ GLAST Initial Value: ‘TBD’

6.2 Special Commands

The following commands must be defined:

GBL_LCLHDR

```
CMD|GBL_LCLHDR|+|||ITOS_CMD||||||||
  “CCSDS Command Local Header Fields”
FLD|GBL_LCLHDR|PH_SEC_HDR|+|UB||0|4|1||1|1||
  “The CCSDS packet header Secondary Header Flag.  Must be
  1 to indicate that a secondary header is present.”

FLD|GBL_LCLHDR|PH_APPID|+|UI||0|5|11|||||
  “The CCSDS packet header Application Process Identifier
(APPID).”
```

FLD|GBL_LCLHDR|PH_PKT_LEN|+|UI||4|0|16||||
  “The CCSDS packet header Packet Length. CCSDS defines this as the
  number of octets in the packet minus 1.”

FLD|GBL_LCLHDR |SH_FUN_CODE|+|UI||6|1|15||||
  “Command Function Code.”

7.0 dbxodb PROGRAM

The dbxodb program processes these transaction records and produces any combination of

- a new operational database
- a set of PDB files (for use by the CMS & DTAS systems)
- a T&C volume II report in HTML
- a T&C volume II report suitable for printing

```
$ dbxodb -help
Usage: dbxodb <options> <files containing dbx transactions>
where <options> are:
  -mkodb, -noodb, -mkhtml, -nohtml, -mkvII, -novII, -mkpdb, -nopdb,
  -mkdbx, and -nodbx
     -- these control which outputs get produced.  The defaults are
        -mkodb, -nohtml, -novII, -nopdb, and -nodbx.
  -odbdir <path>  -- turns on -mkodb and specifies where ODB files get
        written.  Defaults to the current directory.
  -htmldir <path> -- turns on -mkhtml and specifies where html files get
        written.  Defaults to the current directory.
  -urlbase <url>  -- turns on -mkhtml and specifies the URL to the directory
        where html files get written.  If specified, the generated files
        have a <BASE href=...> specification in their <HEAD>.
  -vIIfile <path> -- turns on -mkvII and specifies the basename of the
        output files.  Defaults to './<mission>.<dbvers>.tcvII'; the four
        output files have suffixes '.tp', '.mnem', '.pkt', and '.cmd'.
  -pdbdir <path>  -- turns on -mkpdb and specifies where the PDB files get
        written.  Defaults to the current directory.
  -oldpdb—PDB files generated in pre TRACE era format.
  -dtas   -- PDB files are generated with 84 char records using
        16 character mnemonic names instead of default 12
        used by the Data Trending & Analysis System (DTAS).
  -dbxfile <path> -- turns on -mkdbx and specifies where the dbx transactions
        get written.  Defaults to './<mission>.<dbvers>.yymmddhhmmss.dbx'
  -verbose—turns on additional output messages.  '-verbose -verbose'
        makes things even more verbose!
  -silent  -- turns off everything except error messages.  The default,
        error and warning messages, is equivalent to '-silent -verbose'.
  -mission <mission> -- Names the mission this database is for.
  -version <version> -- Identifies the database version.
  -epochdef <mnemonic name> -- Use <mnemonic name> as the default epoch
        conversion mnemonic on all PKT and FLD records of time and
        date type that don't have an epoch mnemonic defined.
        Default is no default epoch mnemonic is used, and MOC
        telemetry command subsystems and will use the MOC epoch of
```

00:00:00 May 24, 1968.

When conflicting options are specified, the last specification is used.

A Common Database Build Scenario:

```
$ dbxodb -mission trace -version 4.11 -odbdir /home/trace/odb/sparc_SunOS \
-htmldir /home/trace/public_html/tcvol2 \
/home/itos/dbx/*.dbx /home/trace/dbx/trace.sdb4.11
```

-mission trace and -version 4.11 override any initial value in the TLM record for the

ITOS GLOBALS: GBL_MISSION

The name of the current mission.

Destination type	S1
Size	32
Units	
Read only flag	T
Event flag	
Conversion	-none-
Delta Limits	-none-
Limits	-none-
Initial value	“ITOS”

GBL_MISSION is in subsystems: ITOS_DB.

ITOS GLOBALS: GBL_DBVERS

The current database version.

Destination type	S1
Size	64
Units	
Read only flag	T
Event flag	
Conversion	-none-
Delta Limits	-none-
Limits	-none-
Initial value	“GLOBALS”

GBL_DBVERS is in subsystems: ITOS_DB.
mnemonics.

-odbdir /home/trace/odb/sparc_SunOS names the ODB directory. The following files will be created:

```
/home/trace/odb/sparc_SunOS/trace.odb4.11
/home/trace/odb/sparc_SunOS/trace.subsys4.11
/home/trace/odb/sparc_SunOS/trace.tags4.11
/home/trace/odb/sparc_SunOS/trace_cmd.odb4.11
/home/trace/odb/sparc_SunOS/trace_simbuf.odb4.11
/home/trace/odb/sparc_SunOS/traceavlcmdmne4.11
/home/trace/odb/sparc_SunOS/traceavlmnem4.11
/home/trace/odb/sparc_SunOS/traceavlmnemid4.11
/home/trace/odb/sparc_SunOS/traceavlpackid4.11
```

-htmldir /home/trace/public_html/tcvol2 names the HTML directory.

The remaining parameters identify the input transaction files. The /home/itos/dbx/*.dbx files are the MOC-provided transaction files that define MOC global mnemonics, command headers, etc. The transaction files are parsed in the order in which they are included on the DBXODB command line. If there are duplicates, the last one encountered takes precedence.

In addition to the above, DBXODB adds a set of global mnemonics "GBL_PKTCNT_xxxx" where "xxxx" is a 4 digit packet id number for all the packets defined with PKT records. These mnemonics are used by the pktcount display *PAGE*. The user can add records for packets that may be received but have no PKT records so that the pktcount page will display these counts. For example creating a TLM record:

```
TLM|GBL_PKTCNT_0300 |+||ITOS_TEST|U1234|32||||||T|"pkt 300 receive
count"
```

would add a mnemonic for GBL_PKTCNT_0300, which will cause pktcount to display the counts for packet 300. Legal packet ids created by the user may be between 0000 and 2045.

APPENDIX A – ACRONYM LIST

ALG	Analog conversion coefficient
AppID	Application Identification or (APID)
ASCII	American Standard Code for Information Interexchange
CCSDS	Consultative Committee for Space Data Systems
CMD	Command Mnemonic Definition
CMS	Command and Management System
DSC	Discrete
DTAS	Data Trending & Analysis System
FLD	Command Field Definition
GLAST	Gamma-Ray Large Area Space Telescope
GMT	Greenwich Mean Time
HTML	Hypertext Markup Language
IEEE	Institute of Electrical and Electronics Engineers
ITOS	Integrated Test and Operations System
LIM	Telemetry Limit Definition
MAP	Packet Map Attributes
MOC	Mission Operations Center
PDB	Project DataBase
PKT	Telemetry Packet Definition
SEL	Packet Map Selector Definition
SSI	Subsystem Name
STOL	Spacecraft Test and Operation Language
SUB	Command Field Discrete Value Definition
TLM	Telemetry or Global Mnemonic
T&C	Telemetry and Command
UNIX	(Not an acronym)
URL	Universal Resource Locator